Divide & Conquer Algorithms

Ex. Binary Search

Let A be an array of integers (say) with $n = \text{length}[A]$. We will adopt the convention that array indices begin at 1. Thus

$$A[1 \cdots n] = (A[1], \ldots, A[n]).$$

Let $A[p \cdots r]$ denote the subarray

$$A[p \cdots r] = (A[p], \ldots, A[r]).$$

If $p > r$ we understand this to be an empty array.

Assume $A[1 \cdots n]$ is sorted in increasing order (with possible repeated elements.)

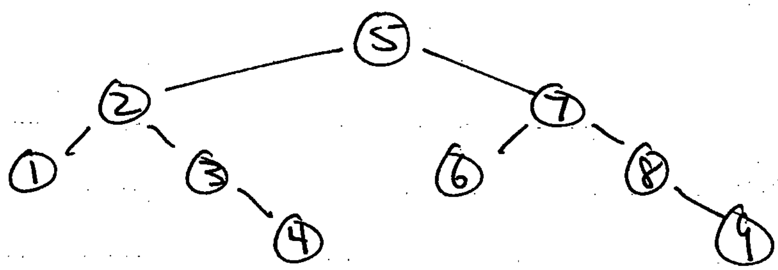
Binary search is a D&C algorithm which locates a given target t in the sub-array $A[p \cdots r]$. If an index i is found such that $A[i] = t$, then i is returned, otherwise 0 is returned.

BinSearch(A, p, r, t) (PRE: $A[p..r]$ SORTED.)

```
if p > r
    return 0
q ← ⌊(p+r)/2⌋
if A[q] = t
    return q
if A[q] < t
    return BinSearch(A, q+1, r, t)
return BinSearch(A, p, q-1, t)
```

Ex. $A = (1, 2, 3, 4, 5, 6, 7, 8, 9)$

1 2 3 4 5 6 7 8 9



THIS BINARY SEARCH TREE REPRESENTS THE ORDER IN WHICH THE TARGET t IS COMPARED TO ELEMENTS OF A.

THE CALL TO BinSearch ON THE FULL ARRAY $A[1..n]$ IS JUST

BinSearch(A, 1, n, t)

THEOREM (CORRECTNESS OF BinSearch)
BinSearch RETURNS EITHER AN INDEX j SUCH THAT $A[j] = t$, OR RETURNS 0 IF NO SUCH j EXISTS.

PROOF
USE INDUCTION ON $m = r - p + 1 = \text{length}[A[p \ldots r]]$.

I. BASE
IF $m=0$ THE SUBARRAY DOES NOT CONTAIN TARGET t. ALSO $m=0$ IMPLIES $r = p-1 < p$, SO THAT 0 IS RETURNED ON LINE 2.

II. (STRONG) INDUCTION
LET $m > 0$ AND ASSUME THAT BinSearch RETURNS THE CORRECT INDEX ON ANY SUBARRAY OF LENGTH LESS THAN m.

NOW $m > 0$ IMPLIES $r > p-1$ WHENCE $r \geq p$. THUS $q = \lfloor \frac{p+r}{2} \rfloor$ (LINE 3) IMPLIES $p \leq q \leq r$.

IF $A[q] = t$ THEN OBVIOUSLY BinSearch RETURNS CORRECT INDEX ON LINE 5.

IF $A[q] < t$ THEN $A[(q+1) \ldots r]$ IS A SUBARRAY OF LENGTH

$$r - (q+1) - 1 = r - q \leq r - p < r - p + 1 = m.$$